CLAIMS

1. A method implemented in a set of instructions executable by a computer that supports floating-point arithmetic operations, the method comprising:

receiving at least one operand represented in fixed-point representation (hereinafter "fixed-point operand"), said fixed-point operand having at least one property selected from a group consisting of (signedness, precision, complexness);

expanding said fixed-point operand into a floating-point representation to obtain a floating-point equivalent;

receiving an instruction comprising an operation to be performed on the fixed-point operand;

performing on the floating-point equivalent, at least one floating-point operation that corresponds to the fixed-point operation, yielding at least one floating-point result; and

reducing, based on a value of said at least one property, said at least one floating-point result generated by the floating-point operation into a corresponding fixed-point result.

2. The method of Claim 1 wherein:

said fixed-point operand is one of at least two fixed-point operands to be used by said instruction; and

said expanding comprises normalization of at least said fixed-point operand if said fixed-point operand has a property different from another operand to be used by said instruction.

3. The method of Claim 1 wherein:

said instruction is to use two operands, with

said fixed-point operand as a first operand, and another fixed-point operand as a second operand; and

said first operand has a property of a first value and the second operand has said property of a second value different from said first value, said expanding comprises normalization of at least one fixed-point operand to have a common value for said property, said common value being one of the first value and the second value.

4. The method of Claim 1 wherein:

the corresponding fixed-point result has said value of at least said one property.

5. The method of Claim 1 further comprising:

determining a property value for the corresponding fixed-point result, based on said at least one property value of the fixed-point operand.

6. The method of Claim 1 comprising:

determining a property value for the corresponding fixed-point result, based on the instruction that was performed on the fixed-point operand.

7. The method of Claim 1 wherein:

the fixed-point representation includes the value of the fixed-point number in memory in floating-point representation.

8. The method of Claim 1 wherein:

the fixed-point representation includes a value in memory to represent the signedness property.

9. The method of Claim 1 wherein:

the fixed-point representation includes a value in memory to represent the complexness property.

10. The method of Claim 1 wherein:

the fixed-point representation uses at least two locations in memory to store a precision of the value.

11. The method of Claim 1 wherein:

the fixed-point representation includes a value in memory to represent a scaling factor for the fixed-point number.

12. The method of Claim 1 further comprising:

storing a precision of the fixed-point operand, prior to performing the floating-point arithmetic operation; and

using the stored precision during reduction of the floating-point result into the corresponding fixed-point result.

13. The method of Claim 1 wherein during reduction of the floating-point result to the corresponding fixed-point result, the method comprises:

using a predetermined storage element to identify a mode of rounding to be performed on the floating-point result, wherein the mode of rounding is one of: round (round-to-nearest), fix (round towards zero), ceil (round towards positive infinity), and floor (round towards negative infinity).

14. The method of Claim 1 wherein during reduction of the floating-point result into the corresponding fixed-point result, the method comprises:

using a predetermined storage element to identify a kind of arithmetic to be performed on the floating-point result, wherein the kind of arithmetic is one of: saturation and modulo.

15.The method of Claim 1 wherein:

said fixed-point representation is hereinafter "first fixed-point representation";

the corresponding fixed-point result is expressed in a second fixed-point representation which is different from the first fixed-point representation; and

the method further comprises using a predetermined storage element to identify a property of the second fixed-point representation.

16.The method of Claim 15 wherein:

said property is precision.

17.The method of Claim 15 wherein:

said property is signedness.

18.The method of Claim 15 wherein:

said property is complexness.

19.The method of Claim 1 wherein during expansion of the fixed-point operands into floating-point equivalents, the method comprises:

detecting that the operands are invalidly scaled and issuing a warning message based on a predetermined storage element.

20.The method of Claim 1 further comprising:

using at least the precision of the fixed-point operand, during emulation of another instruction that uses a result of the fixed-point arithmetic operation.

21.The method of Claim 1 wherein:

the floating-point representation conforms to an IEEE Standard for floating-point arithmetic.

22.The method of Claim 1 wherein the instruction is to be performed on said fixed-point operand and at least an additional floating-point operand, and the fixed-point arithmetic operation is to be performed on the fixed-point operand and said additional floating-point operand, and the method further comprises:

during the act of receiving said floating-point operand, reducing said additional floating-point operand into fixed-point representation, based on the precision of the fixed-point operand.

23.The method of Claim 22 wherein:

reduction of the floating-point operand into said fixed-point representation is based on a property of the fixed-point operand.

24.The method of Claim 23 wherein:

said property is precision.

25.The method of Claim 23 wherein:

said property is signedness.

26.The method of Claim 23 wherein:

said property is complexness.

27.The method of Claim 1 further comprising:

receiving another instruction that indicates a type of said fixed-point operand;

wherein said another instruction comprises a call to a function.

28.The method of Claim 27 wherein:

said function comprises instantiation of an object of a predetermined class, the object comprising said floating-point equivalent and at least one property of said fixed-point operand.

29.The method of Claim 27 wherein:

the fixed-point operand is a real number;

the method further comprises receiving another indication via another function name that a complex number is to be expressed in fixed-point representation; and

on receipt of an imaginary part and a real part of the complex number, expanding each part into a corresponding floating-point equivalent.

30.The method of Claim 1 wherein:

said instruction comprises overloading of an operator normally used to denote said corresponding floating-point operation.

31.A method for using a first program comprising floating-point arithmetic operations to simulate a second program that uses fixed-point arithmetic operations, the method comprising:

a person inserting in the first program an indicator of fixed-point type for each variable that is to be

treated as a fixed-point operand while keeping intact any instructions comprising said fixed-point operand, to obtain the second program; and

a computer emulating each fixed-point arithmetic operation using a corresponding floating-point arithmetic operation during execution of the second program.

32.The method of Claim 31 further comprising:

the computer reducing a result of the corresponding floating-point arithmetic operation into a fixed-point representation; and

the computer storing the fixed-point representation of the result.

33.The method of Claim 31 wherein:

the indicator inserted by the programmer is a name of a function; and

the computer uses the name to identify the type of said each variable as being fixed-point.

34.The method of Claim 31 wherein:

the indicator inserted by the programmer is one of two function names; and

one of the two function names identifies the type as being real and another of the two function names identifies the type as being complex.

35. A computer that supports floating-point arithmetic, the computer comprising:

means (hereinafter "receiving means") for receiving an indication that an operand is to be expressed in a fixed-point representation (hereinafter "fixed-point operand") and for receiving a fixed-point arithmetic

operation to be performed on at least the fixed-point operand;

means, coupled to the receiving means, for expanding the fixed-point operand into a floating-point representation (hereinafter "floating-point equivalent") and for storing a position of a binary point of the fixed-point operand; and

means for performing on the floating-point equivalent a floating-point arithmetic operation that corresponds to the fixed-point arithmetic operation.

36. The computer of Claim 35 further comprising:

means, coupled to the means for performing, for reducing said at least one floating-point result generated by the floating-point arithmetic operation into a corresponding fixed-point result, based on a value of a property of said fixed-point operand.

37. The computer of Claim 36 wherein:

said property is one of signedness, precision and complexness.

38. A method of writing a computer program, the method comprising:

making a function call to identify a variable as being of fixed-point type; and

using the variable in a statement without making another function call to identify the fixed-point type of the variable.

39. The method of Claim 38 further comprising:

identifying a number of properties of the fixed-point type when making the function call.

40. A method implemented in a set of instructions executable by a computer (hereinafter "execution-level language")that supports floating-point arithmetic operations, the method comprising:

receiving at least one operand represented in fixed-point representation (hereinafter "fixed-point operand"), said fixed-point operand having at least one property selected from a group consisting of (signedness, precision, complexness);

expanding said fixed-point operand into a floating-point representation to obtain a floating-point equivalent;

receiving an instruction comprising a fixed-point operation to be performed on the fixed-point operand;

performing on the floating-point equivalent, at least one floating-point operation that corresponds to the fixed-point operation, yielding at least one floating-point result; and

reducing said at least one floating-point result generated by the floating-point operation into a corresponding fixed-point result;

wherein if the fixed-point operation is specified as saturation arithmetic and if said floating-point result is a vector or array, said act of reducing comprises:

setting a maximum value and a minimum value for a fixed-point result to be obtained from reducing the floating-point result generated by the floating-point operation, based on the precision and signedness of at least said fixed-point operand;

replacing any negative numbers in the floating-point result with zero;

replacing any numbers in the floating-point result that are greater than the maximum value with the maximum value;

replacing any numbers in the floating-point result that are less than the minimum value with the minimum value.

41. The method of Claim 40 wherein if a rounding mode is specified as round, the act of reducing further comprises:

subsequent to performance of said act of setting, rounding the floating-point result.

42. A method implemented in a set of instructions executable by a computer that supports floating-point arithmetic operations, the method comprising:

receiving at least one operand represented in fixed-point representation (hereinafter "fixed-point operand");

expanding said fixed-point operand into a floating-point representation to obtain a floating-point equivalent;

receiving an instruction to be performed on the fixed-point operand;

performing on the floating-point equivalent, at least one floating-point operation that corresponds to the fixed-point operation, yielding at least one floating-point result; and

reducing, based on kind of said instruction received, said at least one floating-point result generated by the floating-point operation into a corresponding fixed-point result.

43. The method of Claim 42 wherein:

said instruction includes an operator and said act of reducing is based on said operator.

44. A memory encoded with a plurality of objects, each object representing at least one fixed-point number, each object being encoded in a plurality of locations comprising:

a first location being encoded with a value of a signedness property of said fixed-point number;

a second location being encoded with a value of a complexness property of said fixed-point number;

a plurality of locations being encoded with values of subproperties of a precision property of said fixed-point number;

at least one location being encoded with a floating-point value of said fixed-point number.

45. The memory of Claim 44 wherein each object further comprises:

a third value of a scaling factor of said fixed-point number.

46. The memory of Claim 44 wherein said object further comprises:

a plurality of additional floating-point values;

wherein said values of said properties are identical for each of said additional floating-point values, and said object represents a vector operand.

47. The memory of Claim 44 wherein said precision property comprises:

a number of bits to the left of a point in the fixed-point number as a subproperty.

48. The memory of Claim 47 wherein said precision property further comprises:

a number of bits to the right of said point in the fixed-point number as another subproperty.

49. The memory of Claim 47 wherein said precision property further comprises:
a total number of bits in the fixed-point number as another subproperty.

50. The memory of Claim 44 wherein said precision property further comprises:
a number of bits to the right of said point in the fixed-point number as a subproperty.

51. The memory of Claim 44 wherein said at least one location encoded with said floating-point value of said fixed-point number holds a real component of said fixed-point number, the memory further comprising:
at least one additional memory location for holding a complex component of said fixed-point number.

52. A computer that supports floating-point arithmetic, the computer comprising:
means (hereinafter "receiving means") for receiving at least one operand represented in fixed-point representation (hereinafter "fixed-point operand"), said fixed-point operand having at least one property selected from a group consisting of (signedness, precision, complexness);
means, coupled to said receiving means, for expanding said fixed-point operand into a floating-point representation to obtain a floating-point equivalent;

means for receiving an instruction comprising a fixed-point operation to be performed on the fixed-point operand;

means for performing on the floating-point equivalent, at least one floating-point operation that corresponds to the fixed-point operation, yielding at least one floating-point result;

means for reducing said at least one floating-point result generated by the floating-point operation into a corresponding fixed-point result; and

memory, coupled to each of said means, said memory being encoded with a plurality of objects, at least one object representing at least said fixed-point operand, said object being encoded in a plurality of locations comprising:

a first location being encoded with a value of a signedness property of said fixed-point operand;

a second location being encoded with a value of a complexness property of said fixed-point operand;

a plurality of locations being encoded with values of subproperties of a precision property of said fixed-point operand; and

at least one location being encoded with a floating-point value of said fixed-point operand.